Making sense of EST sequences by CLOBBing them

BMC Bioinformatics

BioMed Central

Research article

Making sense of EST sequences by CLOBBing them

John Parkinson*, David B Guiliano and Mark Blaxter

Address: Institute of Cell, Animal and Population Biology, University of Edinburgh, Edinburgh EH9 3JT, UK

E-mail: John Parkinson* - john.parkinson@ed.ac.uk; David B Guiliano - dguilian@ucl.ac.uk; Mark Blaxter - mark.blaxter@ed.ac.uk

*Corresponding author

Published: 25 October 2002

BMC Bioinformatics 2002, **3**:31

Received: 20 August 2002

Accepted: 25 October 2002

This article is available from: http://www.biomedcentral.com/1471-2105/3/31

Abstract

Background: Expressed sequence tags (ESTs) are single pass reads from randomly selected cDNA clones. They provide a highly cost-effective method to access and identify expressed genes. However, they are often prone to sequencing errors and typically define incomplete transcripts. To increase the amount of information obtainable from ESTs and reduce sequencing errors, it is necessary to cluster ESTs into groups sharing significant sequence similarity.

Results: As part of our ongoing EST programs investigating 'orphan' genomes, we have developed a clustering algorithm, CLOBB (**Cl**uster **o**n the **b**asis of **B**LAST similarity) to identify and cluster ESTs. CLOBB may be used incrementally, preserving original cluster designations. It tracks cluster-specific events such as merging, identifies 'superclusters' of related clusters and avoids the expansion of chimeric clusters. Based on the Perl scripting language, CLOBB is highly portable relying only on a local installation of NCBI's freely available BLAST executable and can be usefully applied to > 95 % of the current EST datasets. Analysis of the *Danio rerio* EST dataset demonstrates that CLOBB compares favourably with two less portable systems, UniGene and TIGR Gene Indices.

Conclusions: CLOBB provides a highly portable EST clustering solution and is freely downloaded from: [http://www.nematodes.org/CLOBB]

Background

Expressed sequence tags (EST) are single pass sequence reads from randomly selected cDNA clones that sample the diversity of genes expressed by an organism [1]. ESTs are a valuable adjunct to whole genome sequencing, as they facilitate gene identification. For organisms where whole genome sequencing is a distant goal, EST analysis is a highly cost-effective gene discovery method. The utility of ESTs is illustrated by the phylogenetic diversity of organisms represented in dbEST, the NCBI's EST database [2].

Random sampling of clones means that redundancy can be expected in EST datasets, even those derived from normalised or subtracted cDNA libraries. Unlike whole genome sequencing, where multiple sequencing of each segment is the norm, ESTs are single pass reads of unverified quality that may contain base-calling and other errors. Additionally an EST may often only provide information on a partial segment of an entire cDNA. Finally, analysis of EST datasets can be overwhelming due to the sheer number of sequences involved.

To address issues of redundancy, quality and data handling, EST clustering can be employed. This involves the

Table 1: Cluster size distribution for the three compared *D. rerio* cluster datasets

Size of Cluster (number of sequences)	*UniGene Build 21 24/08/01*	*TIGR ZGI V.7 07/08/01*	*CLOBB*
1	4169	9914	6848
2	1824	2231	2655
3–4	1953	1956	2321
5–8	1288	1155	1407
9–16	638	506	574
17–32	270	214	230
33–64	123	103	112
65–128	37	41	33
129–256	16	24	24
257–512	12	8	9
513–1024	5	3	2
1025–2048	1	1	0
Total Clusters (from 58,888 sequences)	**10336**	**16156**	**14215**

grouping of ESTs on the basis of sequence similarity into clusters representing putative genes. These groups can then be used to derive consensus sequences that have a higher overall sequence quality and increase the length of transcript that can be assigned. To date a number of different clustering methods have been developed in which ESTs are grouped into a set of "gene indices". These range from simple scripts which run and parse the output of sequence database searches e.g. SEALS [3], INCA [4] and Zymogenetics' REX [5], through more specialised programs such as JESAM [6] and Glaxo's "Dynamic" assembler [7], to programs which rely on non-alignment based algorithms, such as d2_cluster [8]. In addition to these standalone solutions, there are also a number of dedicated database systems such as UniGene [9] and the TIGR Gene Indices [10–12], which create and maintain gene indices derived from entire organismal sets of ESTs.

Our interest in EST clustering arises as part of our involvement in EST projects on 'orphan' genomes. One such project involves a program of gene discovery in parasitic nematodes with the remit of generating ~20,000 ESTs for each of 10 different species of parasitic nematode [13]. To maximise the information derived from these ESTs, for each species of nematode we study a gene index based on the ESTs must be generated. As each dataset may be generated over an extended time period by several different laboratories and we wish to release the information to the public domain as it arises, we required a clustering algorithm that (1) could be run incrementally and (2) which would allow existing clusters to be tracked through subsequent builds. Further, due to the nature of cDNA library construction, the clustering algorithm had to be robust enough to deal with chimeras (clones which arise from the ligation of two unrelated transcripts). In addition, a piece of software was required which was fully accessible (i.e not a pre-built binary) and where parameters could be appropriately set to deal with the nematode datasets. At the beginning of the project, none of the available programs examined were either publicly available in a portable format or fulfilled the aforementioned criteria.

Here we describe a program (CLOBB – **Cl**uster **o**n the **b**asis of **B**LAST) based on the use of BLAST similarity scores [14,15] that achieves these goals. The program is freely available, and is written in the perl scripting language (and is therefore fully customisable). The program depends upon the availability of a locally installed version of BLAST (freely obtainable from [http://www.ncbi.nlm.nih.gov]).

Results and Discussion

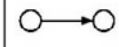
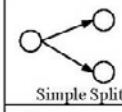
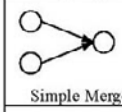
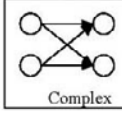
In order to provide a benchmark for the performance of the program, the latest *Danio rerio* (zebrafish) UniGene clustered dataset containing 60,357 sequences (build #21 – 24/08/01) was downloaded and reclustered with CLOBB using the default settings for the tuneable variables outlined in Methods. Vector screening of the sequences downloaded from UniGene revealed 28 sequences with vector sequence. Due to this low level of contamination, it was felt to be more important to maintain consistency with the UniGene build and therefore the original, unmasked sequences were used for clustering. Benchmarking suggests that the time of execution of the algorithm scales exponentially with the number of sequenced involved (results not shown). For the *Danio rerio* dataset, it took a pentium III 750 MHz processor 129 hours and 9 minutes to cluster 60,357 sequences. For our nematode EST datasets, (~20,000 ESTs) the time taken to cluster is clearly acceptable. Of the 387 species represent-

ed in dbEST [2], 370 (> 95%) have fewer than 100,000 ESTs and may therefore be usefully processed by CLOBB. It should be noted, however, that in its present form, CLOBB is unsuitable for much larger datasets (such as the 135,000 *C. elegans* ESTs or 3.8 million human ESTs). Since much of the computing time is spent in reformatting the database for searching with BLAST, future development of the algorithm will investigate reducing the computational overhead associated with this part of the script (potentially by building one initial database containing all sequences, both novel and those previously allocated to a cluster, and performing each search against it).

To compare the performance of our clustering algorithm, we have also downloaded the TIGR gene indices set for *D. rerio* (ZGI Release 7.0 – 07/08/01). In the UniGene process, clusters are initially formed by pair-wise comparison of mRNAs and genomic DNA fragments. ESTs are then added to these clusters providing that such an addition does not lead to the joining of two distinct clusters created in the preceding stage. The TIGR gene indices are built in two stages [16]. Firstly WU-BLAST [14] performs a series of pair-wise sequence comparisons to group all those sequences sharing > = 95% sequence similarity over 40 bp with unmatched overhangs of less than 20 bp. These groups are then subjected to a further round of clustering using the program CAP3 [17,18] to generate a tentative consensus sequence. It should be noted that the TIGR gene index set is based on 95,910 sequences. This discrepancy arises from both the date of clustering (the TIGR and UniGene datasets were obtained at different times) and from the difference in screening methods used by the two protocols. Whilst both methods filter sequences for vector contaminants and polyA tails, rejecting those sequences of <100 bp in length, UniGene also filters out mitochondrial and ribosomal sequences in addition to repetitive elements. The TIGR dataset was therefore filtered for sequences found in the UniGene dataset. In total the TIGR dataset contained 56,888 of the UniGene sequences. The missing 3,469 sequences appear to be derived from EST submissions occurring after 1st August 2001. In the following comparisons between the three cluster datasets, only the 56,888 sequences common to all three were used.

Table 1 compares the number of clusters and their relative abundance for the three *D. rerio* cluster datasets. Both CLOBB and TIGR datasets had more singletons than UniGene (6,848, 9,914 and 4,169 respectively) and increased the overall number of clusters by 40–60% (14,215 and 16,156 vrs 10,336). In comparing the breakdown of cluster size vrs abundance, UniGene has more clusters of larger size than both the CLOBB and the TIGR clusters. It is interesting to note that the TIGR algorithm leads to almost 20% more singletons than the CLOBB algorithm. These

Table 2: Distribution of cluster events

Type of Rearrangement	*CLOBB-UniGene*	*CLOBB-TIGR*	*TIGR-UniGene*
 Equivalent Clusters	6125	10440	6241
 Simple Split	246 → 521	1638 → 4190	107 → 223
 Simple Merge	5838 → 2467	998 → 406	8876 → 3331
 Complex	2006 → 1223	1139 → 1120	932 → 541

comparisons suggest that whilst both the CLOBB and TIGR algorithms appear to be more discriminating than the UniGene clustering algorithm, the CLOBB algorithm appears to be more capable of finding potential matches than the TIGR algorithm. The more inclusive behaviour of the UniGene system probably arises from the inclusion of clone information in the building of clusters. Hence, UniGene clusters may often contain 5' and 3' reads from the same cDNA clone, which do not always overlap. The greater number of singletons in the TIGR clusters compared to those produced by CLOBB is related to both the more stringent overlap cutoff employed by TIGR to reduce the level of chimerism in the initial pair-wise comparisons (40 bases – as opposed to the 30 bases used by CLOBB) and by the incorporation of an assembly process involving CAP3, which can lead to the further splitting of individual clusters.

To compare the datasets more closely we determined how the three sets of clusters are related. We define the relationships in the following ways: equivalent clusters, simple merged (many to one) or split (one to many) clusters and clusters related in complex ways (many to many) (see Table 2). The TIGR and CLOBB clusters are more similar to each other than either is to the UniGene clusters in terms of numbers of equivalent clusters. More TIGR clusters arose from splitting of CLOBB clusters than vice versa, again reflecting the two tier clustering process employed by the TIGR algorithm.

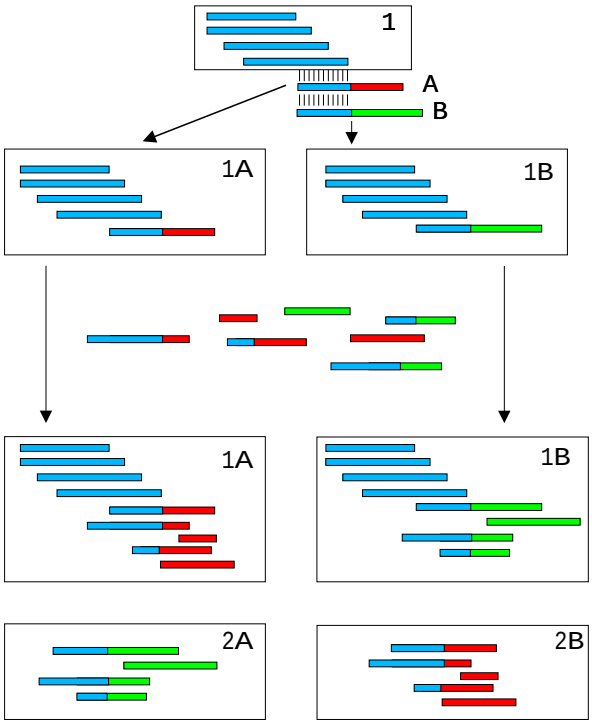


Figure 1
Schematic showing how the history of a cluster can affect its construction. For a given cluster (1), two sequences (A) and (B) show significant identity. Depending upon which sequence is processed first, cluster 1A or cluster 1B can be constructed. Addition of further sequences showing identity to (A) or (B) then leads to the formation of different clusters (1A, 2A) or (1B, 2B) depending on whether cluster 1A or 1B was originally built.

To further understand the differences in behaviour of clustering for the three datasets, we investigated the behaviour of ESTs making up the largest UniGene cluster (Table 3). Cluster ug.2984 encodes a vitellogenin and consists of 1,873 sequences. The CLOBB algorithm splits this cluster into 66 different clusters, and includes an additional 27 sequences. Of these clusters, 33 had one EST member, 20 contained less than 10 sequences; 10 contained 10–100 sequences; and 3 contained 711, 482 and 108 sequences respectively. The TIGR dataset treats ug.2984 as 38 separate clusters and includes an additional 7 sequences (only one of which was in the 27 additional sequences identified in the CLOBB clusters). Of these, 23 had one EST member, 4 contained less than 10 sequences; 7 contained 10–100 sequences; and 4 contained 1,075, 146, 145 and 136 sequences respectively.

To determine how the order in which sequences are added to the CLOBB database may affect clustering, the 1,900 sequences from the CLOBB clusters derived from ug.2984 were reclustered separately using the CLOBB algorithm. This led to the construction of 74 clusters, with only 31 having one EST member (2 of which were not originally found in the ug.2984 dataset). There are now five clusters with greater than 100 sequences, with the largest containing only 425 sequences. This shows, as expected, that clusters formed by the CLOBB algorithm may vary according to the order in which sequences are added. This behaviour is explained by the unidirectional nature of cluster growth. Given that two sequences may share significant similarity to a cluster but not to each other (due to differences in an overlap extending beyond the cluster), further growth of that cluster will depend upon which of the two sequences it encounters first (see Figure 1).

This is a problem common to most clustering algorithms. For example CAP3 was also found to create alternate sets of clusters for ug.2984 by simply altering the order of the sequences in the fasta file it was given. In general such problems tend to be restricted to only a few large clusters and are usually dealt with on a manual basis. To aid this process, we have included a 'supercluster' feature to CLOBB that automatically identifies where such problems may occur. For the ug.2984 cluster used in these analyses, 28 of the 33 clusters, generated in the original CLOBB analysis, containing more than one sequence were tagged as belonging to at least one of four 'superclusters'. Of the remaining five clusters, sequences from one cluster did not show any type II match (see Methods) with any other CLOBB cluster, whilst for the other four clusters, sequences did not have significant ($e < 10^{-5}$) BLAST similarity with sequences from any other CLOBB cluster containing a sequence from the original ug.2984 dataset. Detailed examination of these clusters revealed that their constitutive sequences may have been mis-assigned to ug.2984 by the UniGene algorithm as a result of imperfect clone-read labelling.

By preventing the merging of clusters which despite sharing a common type II match with one sequence, contain within their members sequences that form only type II matches with sequences from the other cluster, CLOBB is able to prevent the merging of two unsuitable clusters via an intermediate chimeric sequence. However, this leads to an increased division of related clusters compared with the TIGR process. From Figure 1, it is clear that these related clusters may in fact contain sequences which are identical to other sequences in another cluster. Since each cluster will be used to predict putative genes (referred to as tentative consensuses (TCs) in the TIGR process), via the use of an assembly program such as PHRAP (Green, P. unpublished) or CAP3, this may result in the prediction of

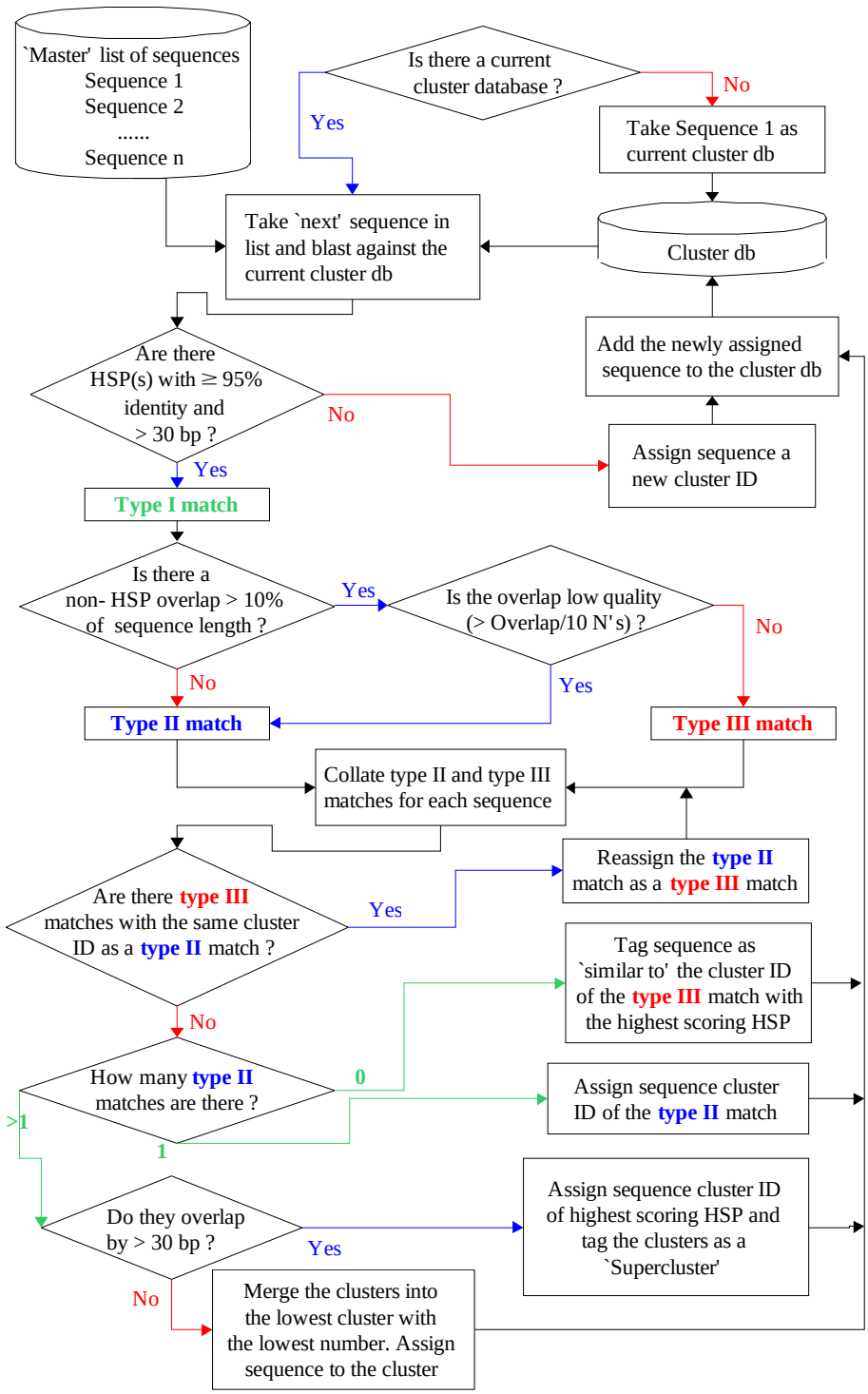


Figure 2
Schematic representation of the cluster process. For a further explanation of the clustering process see text.

Table 3: Detailed analysis of UniGene cluster ug.2984

	UniGene	*TIGR*	*CLOBB-α*	*CLOBB-β*
Total sequences in clusters containing at least one sequence derived from ug.2984	1873	1880	1900	1900
Total clusters	1	38	66	74
Clusters with > 100 seqs (sizes)	1	4 (1075, 146, 145, 136)	3 (711,482,108)	5 (425, 219, 214, 203, 143)
Clusters with only one sequence (singletons)	0	23	33	31

Table 4: Post cluster consensus assembly using CAP3 of CLOBB clusters derived from ug.2984

Cluster(s) assembled	*Total number of clusters used in assembly (sequences)*	*Number of contigs built from > 1 sequence (A)*	*Number of singlets produced in assembly (B)*	*Total predicted tentative consensuses (A) + (B) + number of singletons*
CLOBB pooled clusters	1 (1900)	20	23	43
CLOBB-α individual clusters	33 (1867)	46	11	90
CLOBB-β individual clusters	43 (1869)	56	14	101
TIGR predictions for the same 1900 sequences	N/A	30	19	49

common consensuses for different clusters and hence lead to an overprediction in the total number of putative genes. To investigate the potential scale of overprediction, the CLOBB clusters created for ug.2984 were assembled into consensuses using the program CAP3 (see Table 4). Both sets of CLOBB clusters predict almost double the number of TCs (90 and 101 respectively) as the TIGR process (49). However, when the clusters are pooled into a single cluster containing all 1,900 sequences and subjected to assembly, the number of TCs is actually fewer (43) than predicted by the TIGR algorithm. These figures indicate that by avoiding the merging of related clusters, the CLOBB process overpredicts the total number of putative genes. For this reason, we have included a 'superclustering' feature into the algorithm. Post-CLOBB sequence assembly can then use this 'supercluster' information to merge related clusters prior to the assembly step and hence reduce the number of TCs.

In terms of maintaining continuity between builds, the UniGene clusters are renumbered after comparison with the previous build. However, it should be noted that sequences assigned to a cluster may change with subsequent builds and that cluster identifiers may disappear (typically when two clusters merge). For the TIGR clusters, identifiers which change through mergers or splits are kept in their database. In CLOBB merges and splits are recorded and cluster membership is tracked for each EST.

Conclusions

These results demonstrate the uncertain nature of automatically derived sequence clusters in the absence of trace quality information. The three different methods compared here (Table 5) embody different philosophies of cluster discovery. For our purposes, the CLOBB algorithm is a robust and useful tool to identify clusters of EST sequences that share sequence similarity. Unlike the TIGR assembly process it is not intrinsically linked to a local database system and is therefore easily portable requiring only a local installation of the freely available NCBI BLAST algorithm [14,15]. Furthermore, as it automatically increments between subsequent builds, it is able to record 'historic' events such as 'superclusters' and mergers. Although CLOBB does not attempt to derive consensus sequences, it is a relatively trivial task to post-process the clusters using contig assembly tools such as CAP3 or PHRAP. Despite being unsuitable for very large datasets, CLOBB is nonetheless recommended as a useful clustering solution for more than 95 % of the available species datasets in NCBI dbEST.

CLOBB is freely available with POD documentation from our website [http://www.nematodes.org/CLOBB].

Table 5: Summary of features of the three cluster methods examined

Feature	*UniGene*	*TIGR*	*CLOBB*
Underlying Clustering Method	megaBLAST	WU-BLAST & CAP3	NCBI BLAST
Stringency	Dependent on stage of clustering	Very High	High
		>= 95% identity over > 40 bp	>= 95% identity over 30 bp
Overlap allowed	N/A	< 20 bp	< 10% of sequence length
			Those with > 10% of sequence length are allowed if they contain > 10% unassigned bases
Clusters are always contiguous?	No	Yes	Yes
Dealing with potential chimeric clusters	Initial clustering performed with gene sequences – merging of these initial distinct clusters rejected	CAP3 does not include identified chimeric sequences	Definition of type III matches and 'superclusters' prevents chimeric sequences from merging unsuitable clusters.
Continuity (addition of new sequences)	New builds are compared with previous builds	Post processing	Incremental within algorithm
Historical information	Availability of previous builds	Notes showing retirement of clusters	'superclusters' and merge events can be tagged
Portability and adapatibility	Low	Low	High
Ease of retention of manual curation	Medium	Medium	High

Methods

Computational

Calculations were performed on an Intel pentium III 750 Mhz processor running Red Hat Linux 6.2 with perl version 5.005. The script was written in the perl scripting language and uses NCBI BLAST version 2.1.2.

The algorithm

CLOBB is an iterative clustering method – that is the cluster database increases by the addition of new sequences. The program follows the schematic outlined in Figure 2. Firstly all ESTs to be clustered are placed in a common directory in FASTA format. When the program is run, a 'master' list is made of all the files in this directory. If a previous cluster build has been performed, it is read to determine the number of the last identified cluster. The first EST is then compared using BLASTN to the current cluster database. The BLAST output is parsed for high-scoring segment pairs (HSPs). For all HSPs with an identity of >= 95% and length > 30 bp, the subject sequence is recorded as a type I match. The next stage of the process goes through the list of type I matches to identify any problems associated with the match. This is achieved by parsing the beginning and end positions of the query and subject sequences from the BLAST output. If these positions overlap beyond the HSP by more than 30 bases (i.e. the HSP does not extend through the full overlap of the sequences), a further check is performed to ensure that this is not due to the presence of poor quality sequence (determined by the number of bases assigned 'N' in the overlap regions). Type I matches which do not have high quality overlaps of more than 30 bases beyond the HSP are designated as type II matches. Other type I matches which possess high quality overlaps of greater than 30 bases which are not part of a HSP are designated as type III matches.

The next stage of cluster assignment then involves checking through the lists of type II and type III matches to ensure that no conflicts arise. Given a cluster in which some members are type II matches, if there are other members of the same cluster which have been designated as type III matches, then this indicates that the query sequence matches some but not all members of a cluster and is therefore assigned a new cluster number. The inclusion of this feature in the algorithm prevents the rapid expansion of chimeric clusters and can result in a splitting of related sequences into many different (related) clusters. However, when such events occur, the program catalogues the clusters involved, identifying them as 'similar to' the type III match for subsequent post-process analysis (typically performed by manual curation).

Another complication occurs when two or more type II matches arise from different clusters. Firstly the BLAST

output is re-analysed to determine whether the HSPs of the matches occur in overlapping regions. If they do not, the query effectively links the clusters and they are merged into the cluster with the lowest index – a separate note is recorded to indicate that such a merge operation has occurred. If they do overlap, this may indicate that they are either alternatively spliced variants of one gene or closely related members of a gene family, and the query sequence is assigned the cluster number of the type II match with which it had the highest BLAST score, providing that said cluster did not contain a type III match, and an annotation added to indicate that these clusters may be members of a 'supercluster'. Once the query sequence has been assigned to a cluster, it is added to the growing cluster database which is then reformatted to allow the next search.

From the above schema it is apparent that the script contains a number of tuneable variables such as percentage identity in overlap, minimum length of HSP and maximum allowable non-HSP overlap. CLOBB is designed to use EST sequences downloaded from dbEST. Access to quality data from sequence chromatograms would make it possible to use more accurate measures than simply the number of N's in overlapping regions to determine regions of poor quality. Due to the portability and readability of the perl scripting languages, such features would be easy to introduce.

Authors' Contributions

JP developed CLOBB, incorporated many new features, performed the analyses on the *Danio rerio* dataset and drafted the manuscript. DG wrote the original script on which CLOBB is based. MB conceived of the program and participated in its design. All authors read and approved the final manuscript.

Acknowledgements

We would like to thank James Callahan (Smith College, MA, USA) for assistance with early versions of the scripting, and the Blaxter Nematode Genomics lab for helpful discussions. This work was funded by the Wellcome Trust and the UK Medical Research Council.

References

1. Adams MD, Kelley JM, Gocayne JD, Dubnick M, Polymeropolous MH, Xiao H, Merril CR, Wu A, Olde B, Moreno RF, *et al*: **Complementary DNA sequencing: Expressed sequence tags and human genome project.** *Science* 1991, **252:**1651-1656
2. Boguski MS, Lowe TM, Tolstoshev CM: **dbEST–database for "expressed sequence tags".** *Nat Genet* 1993, **4:**332-333
3. Walker DR, Koonin EV: **SEALS: A System for Easy Analysis of Lots of Sequences.** *Int. Sys. Mol. Biol.* 1997, **5:**333-339
4. Graul RC, Sadée W: **Evolutionary relationships among proteins probed by an iterative neighborhood cluster analysis (INCA). Alignment of bacteriorhodopsins with the yeast sequence YRO2.** *Pharm Res* 1997, **14:**1533-1541
5. Yee DP, Conklin D: **Automated clustering and assembly of large EST collections.** *Proc Int Conf Intell Syst Mol Biol* 1998, **6:**203-211
6. Parsons JD, Rodriguez-Tomé P: **JESAM: CORBA software components to create and publish EST alignments and clusters.** *Bioinformatics* 2000, **16:**313-325
7. Gill RW, Hodgman TC, Littler CB, Oxer MD, Montgomery DS, Taylor S, Sanseau P: **A new dynamic tool to perform assembly of expressed sequence tags (ESTs).** *Comput Appl Biosci* 1997, **13:**453-457
8. Burke J, Davison D, Hide W: **d2_cluster: A validated method for clustering EST and full-length cDNA sequences.** *Genome Res* 1999, **9:**1135-1142
9. Boguski MS, Schuler GD: **ESTablishing a human transcript map.** *Nat Genet* 1995, **10:**369-371
10. Adams MD, Kerlavage AR, Fleischmann RD, Fuldner RA, Bult CJ, Lee NH, Kirkness EF, Weinstock KG, Gocayne JD, White O, *et al*: **Initial assessment of human gene diversity and expression patterns based upon 83 million nucleotides of cDNA sequence.** *Nature* 1995, **377: Supplement:**3-174
11. Sutton GG, White O, Adams MD, Kerlavage AR: **TIGR Assembler: A new tool for assembling large shotgun sequencing projects.** *Gen Sci Technol* 1995, **1:**9-19
12. White O, Kervalage AR: **TDB: new databases for biological discovery.** *Meths Enzymol* 1996, **266:**27-40
13. Parkinson J, Whitton C, Guiliano D, Daub J, Blaxter M: **200 000 nematode expressed sequence tags on the net.** *Trends Parasitol* 2001, **17:**394-396
14. Altschul SF, Gish W, Miller W, Myers MW, Lipman DJ: **Basic local alignment search tool.** *J Mol Biol* 1990, **215:**403-10
15. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ: **Gapped BLAST and PSI-BLAST: a new generation of protein database search programs.** *Nucleic Acids Res* 1997, **25:**3389-3402
16. Liang F, Holt I, Pertea G, Karamycheva S, Salzberg SL, Quackenbush J: **An optimized protocol for analysis of EST sequences.** *Nucleic Acids Res* 2000, **28:**3657-3665
17. Huang X: **An improved sequence assembly program.** *Genomics* 1996, **33:**21-31
18. Huang X, Madan A: **CAP3: A DNA Sequence Assembly Program.** *Genome Res* 1999, **9:**868-877